This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims

1. (Previously Presented) A method implemented in circuitry, comprising:

accessing a program comprising a plurality of instructions including at least one no operation (NOP) instruction;

determining one instruction in the program preceding a determined NOP instruction whose movement forward to replace the determined NOP instruction will not result in data not being available when needed; and

replacing the determined NOP instruction with the determined instruction preceding the determined NOP instruction.

2. (Previously Presented) The method of claim 1, further comprising:

deleting one NOP instruction in the program that is not needed to provide a processing delay to ensure the data is available to at least one dependent instruction without moving a non-NOP instruction.

3. (Previously Presented) The method of claim 1, further comprising:

deleting at least one NOP instruction in the program that is not needed to provide the processing delay to ensure the data is available to at least one dependent instruction; and

after deleting the at least one instruction, replacing at least one NOP instruction with one determined instruction, preceding the at least one NOP instruction, whose movement forward to replace the determined NOP instruction will not result in data not being available when needed.

4. (Original) The method of claim 3, further comprising:

performing an additional iteration of deleting at least one instruction and then replacing the at least one NOP instruction in response to replacing at least one NOP instruction.

5. (Original) The method of claim 1, wherein the instructions in the program comprise assembly language instructions coded by a developer.

6. (Previously Presented) The method of claim 2, wherein deleting NOP instructions in the program further comprises accessing and processing each NOP instruction by:

determining whether the accessed NOP instruction is needed to delay processing of one dependent instruction following the accessed NOP instruction to ensure that data is available to the dependent instruction accessing the data; and

deleting the accessed NOP instruction in response to determining that the NOP instruction is not needed to ensure that data is available to the dependent instruction accessing the data..

7. (Previously Presented) The method of claim 6, wherein determining whether the accessed NOP instruction is needed to delay processing of one dependent instruction further comprises:

identifying instructions preceding the NOP instruction that have a delay in writing the results[.]; and

identifying dependent instructions following the NOP instruction that are dependent on an availability of data from the identified instructions having the delay in writing the results.

8. (Previously Presented) The method of claim 1, wherein the determining of one instruction in the program to move forward comprises determining one instruction whose movement forward to replace the determined NOP instruction will not result in data not being available to one dependent instruction following the NOP instruction.

9. (Original) The method of claim 8, wherein the one previous instruction comprises a preceding instruction closest to the accessed NOP instruction in the program.

10. (Previously Presented) The method of claim 8, further comprising:

deleting at least one NOP instruction not needed to ensure that data accessed by the dependent instruction is available to the dependent instruction, wherein the operations of replacing accessed NOP instructions with previous non-NOP instructions are performed after

deleting NOP instructions not needed to ensure that data accessed by the dependent instruction is available.

11. (Previously Presented) The method of claim 1, wherein the determined instruction is further not a branch target instruction.

12. (Original) The method of claim 1, wherein the program instructions are for execution by an engine in a multiprocessor engine.

13. (Previously Presented) A system for processing a plurality of instructions including at least one no operation (NOP) instruction, comprising:

a processor;

a code optimizer executed by the processor to perform operations, the operations comprising:

access the program;

determine one instruction in the program preceding a determined NOP instruction whose movement forward to replace the determined NOP instruction will not result in data not being available when needed; and

replace the determined NOP instruction with the determined instruction preceding the determined NOP instruction.

14. (Previously Presented) The system of claim 13, wherein the operations further comprise:

delete one NOP instruction in the program that is not needed to provide the processing delay to ensure the data is available to at least one dependent instruction without moving a non-NOP instruction.

15. (Previously Presented) The system of claim 13, wherein the operations further comprise:

delete at least one NOP instruction in the program that is not needed to provide the processing delay to ensure the data is available to at least one dependent instruction; and

after deleting the at least one instruction, replace at least one NOP instruction with one determined instruction, preceding the at least one NOP instruction, whose movement forward to replace the determined NOP instruction will not result in data not being available when needed.

16. (Previously Presented) The system of claim 15, wherein the operations further comprise:

perform an additional iteration of deleting at least one instruction and then replacing the at least one NOP instruction in response to replacing at least one NOP instruction.

17. (Original) The system of claim 13, wherein the instructions in the program comprise assembly language instructions coded by a developer.

18. (Previously Presented) The system of claim 14, wherein the operation to delete NOP instructions in the program further comprises accessing and processing each NOP instruction to:

determine whether the accessed NOP instruction is needed to delay processing of one dependent instruction following the accessed NOP instruction to ensure that data is available to the dependent instruction accessing the data; and

delete the accessed NOP instruction in response to determining that the NOP instruction is not needed to ensure that data is available to the dependent instruction accessing the data.

19. (Previously Presented) The system of claim 18, wherein the operation to determine whether the accessed NOP instruction is needed to delay processing of one dependent instruction further performs:

identify instructions preceding the NOP instruction that have a delay in writing the results; and

identify dependent instructions following the NOP instruction that are dependent on an availability of data from the identified instructions having the delay in writing the results.

20. (Previously Presented) The system of claim 13, wherein the determining of one instruction in the program to move forward comprises determining one instruction whose

movement forward to replace the determined NOP instruction will not result in data not being available to one dependent instruction following the NOP instruction.

21. (Previously Presented) The system of claim 20, wherein the one previous instruction comprises a preceding instruction closest to the accessed NOP instruction in the program.

22. (Previously Presented) The system of claim 13, wherein the operations further comprise:

delete at least one NOP instruction not needed to ensure that data accessed by a dependent instruction is available to the dependent instruction, wherein the operations of replacing accessed NOP instructions with previous non-NOP instructions are performed after deleting NOP instructions not needed to ensure that data accessed by the dependent instruction is available.

23. (Previously Presented) The system of claim 13, wherein the determined instruction is further not a branch target instruction.

24. (Previously Presented) An article of manufacture comprising at least one of hardware logic and a computer storage medium having code that is executed to perform operations, the operations comprising:

access a program comprising a plurality of instructions including at least one no operation (NOP) instruction;

determine one instruction in the program preceding a determined NOP instruction whose movement forward to replace the determined NOP instruction will not result in data not being available when needed; and

replace the determined NOP instruction with the determined instruction preceding the determined NOP instruction

.

25. (Previously Presented) The article of manufacture of claim 24, wherein the operations further comprise:

delete one NOP instruction in the program that is not needed to provide a processing delay to ensure the data is available to at least one dependent instruction without moving a non-NOP instruction.

26. (Previously Presented) The article of manufacture of claim 24, wherein the operations further comprise:

delete at least one NOP instruction in the program that is not needed to provide a processing delay to ensure the data is available to at least one dependent instruction; and

after deleting the at least one instruction, replace at least one NOP instruction whose movement forward to replace the determined NOP instruction will not result in data not being available when needed.

27. (Previously Presented) The article of manufacture of claim 26, wherein the operations further comprise:

perform an additional iteration of deleting at least one instruction and then replacing the at least one NOP instruction in response to replacing at least one NOP instruction.

28. (Original) The article of manufacture of claim 24, wherein the instructions in the program comprise assembly language instructions coded by a developer.

29. (Previously Presented) The article of manufacture of claim 25, wherein the deleting of the NOP instructions in the program further accesses and processes each NOP instruction to:

determine whether the accessed NOP instruction is needed to delay processing of one dependent instruction following the accessed NOP instruction to ensure that data is available to the dependent instruction accessing the data; and

delete the accessed NOP instruction in response to determining that the NOP instruction is not needed to ensure that data is available to the dependent instruction accessing the data.

30. (Previously Presented) The article of manufacture of claim 29, wherein the operation to determine whether the accessed NOP instruction is needed to delay processing of one dependent instruction further performs:

identify instructions preceding the NOP instruction that have a delay in writing the results; and

identify dependent instructions following the NOP instruction that are dependent on a availability of data from the identified instructions having the delay in writing the results.

31. (Previously Presented) The article of manufacture of claim 24, wherein the determining of one instruction in the program to move forward comprises determining one instruction whose movement forward to replace the determined NOP instruction will not result in data not being available to one dependent instruction following the NOP instruction.

32. (Original) The article of manufacture of claim 31, wherein the one previous instruction comprises a preceding instruction closest to the accessed NOP instruction in the program.

33. (Previously Presented) The article of manufacture of claim 31, wherein the operations further comprise:

delete at least one NOP instruction not needed to ensure that data accessed by the dependent instruction is available to the dependent instruction, wherein the operations of replacing accessed NOP instructions with previous non-NOP instructions are performed after deleting NOP instructions not needed to ensure that data accessed by the dependent instruction is available.

34. (Previously Presented) The article of manufacture of claim 24, the determined instruction is further is not a branch target instruction.

35. (Original) The article of manufacture of claim 24, wherein the program instructions are for execution by an engine in a multiprocessor engine.

36. (New) The method of claim 1, wherein determining one instruction in the program preceding the determined NOP instruction whose movement forward to replace the determined NOP instruction will not result in data not being available when needed comprises determining whether the instruction to move forward causes the data needed by one dependent instruction to be written in fewer cycles such that the number of cycles between a writing instruction and the dependent instruction are not sufficient to guarantee that the written data will be available to the dependent instruction.